DOCUMENT-IDENTIFIER: US 20010052108 A1

TITLE: SYSTEM, METHOD AND ARTICLE OF MANUFACTURING FOR A DEVELOPMENT ARCHITECTURE FRAMEWORK

---------- KWIC ---------

Abstract Paragraph - ABTX (1):

A method, system, and article of manufacture are provided for designing, implementing, and maintaining a development architecture framework. The present invention manages information that supports a project being carried out by a development architecture framework. Further, security of the development architecture framework is handled by defining security requirements. Thereafter, the development architecture framework is audited to ensure that the security requirements are met. Quality of the project being carried out by the development architecture framework is ensured by obtaining measurements relating to predetermined criterion of the project. Such measurements are statistically analyzed. Next, personnel are trained based on the statistical analysis in order to improve the quality of the project. Throughout the process, the project being carried out by the development architecture framework is managed by generating a plan to carry out the project, scheduling a timeline for executing the plan, tracking the execution of the plan, and **reporting** information uncovered during tracking. An environment in which the project is carried out by the development architecture framework is also managed. The delivery of components of the project are coordinated in a selected order. Further, problems that occur during the delivery of the components of the project are rectified and updated support information is maintained during the delivery.

Application Filing Date - APD (1):

19990831

Summary of Invention Paragraph - BSTX (14):

[0010] Throughout the process, the project being carried out by the development architecture framework is managed by generating a plan to carry out the project, scheduling a timeline for executing the plan, tracking the execution of the plan, and **reporting** information uncovered during tracking.

Detail Description Paragraph - DETX (31):

[0056] The development of graphical user interfaces began to turn this procedural programming arrangement inside out. These interfaces allow the user, rather than program logic, to drive the program and decide when certain actions should be performed. Today, most personal computer software accomplishes this by means of an event loop which **monitors** the mouse, keyboard, and other sources of external events and calls the appropriate parts of the programmer's code according to actions that the user performs. The programmer

no longer determines the order in which events occur. Instead, a program is divided into separate pieces that are called at unpredictable times and in an unpredictable order. By relinquishing control in this way to users, the developer creates a program that is much easier to use. Nevertheless, individual pieces of the program written by the developer still call libraries provided by the operating system to accomplish certain tasks, and the programmer must still determine the flow of control within each piece after it's called by the event loop. Application code still "sits on top of" the system.

Detail Description Paragraph - DETX (41):
[0066] To date, Web development tools have been limited in their ability to create dynamic Web applications which span from client to server and interoperate with existing computing **resources**. Until recently, HTML has been the dominant technology used in development of Web-based solutions. However, HTML has proven to be inadequate in the following areas:

Detail Description Paragraph - DETX (74):
[0099] Relying on the Business Integration Methodology and its project organization guidelines (0940--Organize Project **Resource** Task Package), the following should be prepared:

Detail Description Paragraph - DETX (79):
[0104] Accountability--How and by whom the performance will be **measured**

Detail Description Paragraph - DETX (99):
[0124] The Information Management team is responsible for ensuring that the project's knowledge capital and information **resources** are managed effectively. This includes:

Detail Description Paragraph - DETX (103):
[0128] Information Management encompasses Repository management, but generally has a broader scope than merely the repository contents, because most repositories are not capable of holding all the information **resources** of a project. It is, for example, common to have key project information reside in a combination of repositories, teamware **databases,** flat files, and paper documents. It is the Information Management team's responsibility to ensure consistency across all these formats. The responsibilities of the Information Management team therefore cover:

Detail Description Paragraph - DETX (109):
[0134] In addition to managing the information for the System Building team, the Information Management team must also manage the information **resources** of the other management processes--quality management, environment management, and project management.

Detail Description Paragraph - DETX (112):
[0137] **Database** administration--this is part of the Architecture team responsibilities

Detail Description Paragraph - DETX (135):
[0160] Program Performance **Reporting**

Detail Description Paragraph - DETX (136):
[0161] **Resource** Management

Detail Description Paragraph - DETX (142):
[0167] **Resource** consumption

Detail Description Paragraph - DETX (152):
[0177] To **optimize** processes

Detail Description Paragraph - DETX (157):
[0182] **Measuring and monitoring** progress using established processes to ensure that a capability release is delivered on time, within budget, and that it meets or exceeds expectations.

Detail Description Paragraph - DETX (159):
[0184] Ensuring that **resources** are used effectively across projects for the release.

Detail Description Paragraph - DETX (188):
[0213] **Database** administration

Detail Description Paragraph - DETX (189):
[0214] **Database** tuning

Detail Description Paragraph - DETX (244):
[0269] A vast amount of information is generated within the development environment, which needs to be carefully managed (for example, design documentation, application code, media content, test plans and test data). Information Management generally involves Repository Management, Folder Management and, where applicable, Object Management and Media Content Management. Since a number of teams rely on the service provided by the information management team, it is important that the level of service to be provided be chosen carefully, documented, and communicated. The arrangement should take the form of a **Service Level Agreement (SLA)**. Such an **SLA** typically defines how quickly a new data element is created and how repository changes are communicated. More generally it defines the division of responsibilities between the information management team and the other project teams at a

detailed level.

Detail Description Paragraph - DETX (247):
[0272] **Monitoring** and controlling update activities in the repository

Detail Description Paragraph - DETX (255):
[0280] As many repositories do not provide sufficient versioning functionality, it is common to have more than one repository on large projects. Typically, there may be one repository for development, one for system test, and one for production. This allows better control, but also requires significant **resources** to move repository objects from the development environment to the system test environment. By merging the development and system test repositories, the medium-sized project has a potential for productivity gains. If these gains are to be realized, great care must be taken when making corrections during system test. As a common repository is shared, any error analysis involving repository objects must take into account the possibility that these objects could have changed since the previous migration to system test. This situation can be managed by meticulously maintaining a comprehensive change log.

Detail Description Paragraph - DETX (264):
[0289] Analysis, **reporting,** and querying

Detail Description Paragraph - DETX (270):
[0295] Requests for data element changes can be forwarded using a **database** or paper-based system. Based on functional and technical knowledge, the repository administrator evaluates the requests and may involve other teams to make appropriate decisions. The **database** used to request data element changes during design and programming should be separate from the project's change request **database**. This will simplify and speed up the change process. When data elements have to be changed during system test, however, the impact can be much greater, and the regular change request **database** should be used.

Detail Description Paragraph - DETX (271):
[0296] Whenever a data element is changed, impact analysis must be performed to understand the side-effects. Where-used **reports** are useful to determine these side-effects. The repository manager must be able to obtain the list of direct references and the list of all components affected indirectly (transitive closure). In the latter case, a message based on a record containing a group, which makes reference to a changed data element is considered to be indirectly affected by the change. When adding a data element, no functional equivalent must exist, because redundancy creates difficulties for impact analysis and future maintenance.

Detail Description Paragraph - DETX (273):
[0298] The objects related to dialog definitions, **reports,** messages, and so forth, are usually maintained by the designers and programmers. When the

dialogs and **report** programs are tested, approved, and ready to be promoted to the system test environment, the related objects must be locked. This is the responsibility of the Repository Management team.

Detail Description Paragraph - DETX (276):
[0301] Detailed, project-specific standards should exist for defining repository objects. These standards can form the basis for a repository validation program, which can run through the entire repository and **report** on detected deviations from standards. In some cases, this program can also enforce the standard.

Detail Description Paragraph - DETX (277):
[0302] Mass changes to the repository can be performed when the validation **reports** show the occurrence of many standards violations that follow a common pattern. This may occur in cases where:

Detail Description Paragraph - DETX (282):
[0307] Analysis, **Reports,** and Queries

Detail Description Paragraph - DETX (283):
[0308] Certain **reports** should be run daily, such as the list of new data elements or modified data elements. These **reports** can serve as an audit trail of changes and can be used to communicate changes to the entire team. Procedures should specify which **reports** are run daily and what their distribution should be.

Detail Description Paragraph - DETX (285):
[0310] When supporting specific kinds of repository analysis, the Repository Management team can provide custom **reports** or ad hoc queries that satisfy particular needs.

Detail Description Paragraph - DETX (357):
[0382] To fine-tune the development process, the important quality attributes must be **measured**. Sample metrics include:

Detail Description Paragraph - DETX (385):
[0410] Identify the **resource** allocation process

Detail Description Paragraph - DETX (387):
[0412] Identify how the effort will be **monitored**

Detail Description Paragraph - DETX (392):
[0417] Allocate **resources** and scheduling

Detail Description Paragraph - DETX (393):
[0418] **Monitor** effort

Detail Description Paragraph - DETX (395):
[0420] While maintaining quality at a program level, the Quality Management team must liaise with each of the organizational units within the development environment in order to **monitor** the quality management processes within these units.

Detail Description Paragraph - DETX (399):
[0424] For more details, refer to Consistently Delivering Value: The CMM--How to Help Your Project **Measure** Up.

Detail Description Paragraph - DETX (414):
[0439] 0710--Obtain and Deploy **Resources**

Detail Description Paragraph - DETX (430):
[0455] Project Management focuses on providing specific deliverables through balanced management of scope, quality, effort, risk, and schedule. Project Management processes follow a cycle of planning the project's execution, organizing its **resources,** and controlling its work. The Project Management team oversees all other teams within the development environment.

Detail Description Paragraph - DETX (434):
[0459] 0940--Organize Project **Resources**

Detail Description Paragraph - DETX (506):
[0531] All processes that are performed by the Environment management team must be documented in a centralized **database** that allows quick and easy reference.

Detail Description Paragraph - DETX (509):
[0534] Management of **Service Level Agreements (SLAs)**

Detail Description Paragraph - DETX (512):
[0537] **Service Level Agreements**

Detail Description Paragraph - DETX (513):
[0538] In order to plan and organize the development work appropriately, a **Service Level Agreement (SLA)** must be in place between the Service Management group (typically part of the Environment Management team) and the developers. As with all other components of the development environment, this agreement should be kept simple. It should specify the following:

Detail Description Paragraph - DETX (517):
[0542] How the Environment Management team will notify developers of environment changes such as changes to **databases** and common technical modules.

Detail Description Paragraph - DETX (518):
[0543] Specifications of service levels should be precise and the service must be measurable. The **SLA** should also specify how to **measure** this service (for example, system response times, request service times, backup frequencies). In addition, the **SLA** must be managed. It may have to be modified as the environment changes, and it must be reviewed with developers on a regular basis to see if the service level is adequate.

Detail Description Paragraph - DETX (532):
[0557] Additional entries in a "technical hints" **database**

Detail Description Paragraph - DETX (534):
[0559] Efficient searches in the Help Desk **database** can, in some cases, be greatly facilitated by extending the basic functionality of the Help Desk tool. This can be achieved, for example, by adding a smart word search capability on top of the Help Desk history **database**.

Detail Description Paragraph - DETX (537):
[0562] To manage communication with external vendors, a contacts **database** with the following information is useful:

Detail Description Paragraph - DETX (551):
[0576] Defining the **SLA**, with its specific, measurable criteria, is the basis for continuous improvement. The continuous improvement effort may focus on providing the same level of service with fewer **resources**, or on providing better service. An important part of quality management is ensuring that the Environment Management team understands the key performance indicators for service delivery, that these indicators are **monitored**, and that all personnel are adequately equipped with the tools and training to fill their responsibilities. While the entire team is responsible for delivering quality, the responsibility for Quality management should be assigned to a specific individual on the Environment Management team.

Detail Description Paragraph - DETX (555):
[0580] **Monitoring**

Detail Description Paragraph - DETX (561):
[0586] Reorganization of **databases**, including the repository

Detail Description Paragraph - DETX (562):
[0587] Rerunning of **database** statistics

Detail Description Paragraph - DETX (568):
[0593] **Monitoring**

Detail Description Paragraph - DETX (569):
[0594] The Environment Management team must systematically **monitor** the development environment to ensure that it is stable, provides adequate response times, and satisfies the needs of the developers. This **monitoring** involves looking at trends and extrapolating them to anticipate problems with disk capacity, system performance, network traffic, and so forth.

Detail Description Paragraph - DETX (582):
[0607] In the development environment, it may be possible to outsource certain Systems Management tasks. For example, the LAN supplier may be willing to take responsibility for LAN support, upgrades, and so on. Similarly, an existing data processing center may be willing to take responsibility for host operations. Such agreements are very beneficial and make it possible to use project team members more effectively. However, outsourcing the development environment carries a risk, which can be mitigated by defining a **Service Level Agreement** with the provider. This will generally be very similar to the **SLA** established between the Environment Management team and the developers. One important difference is that punitive **measures** (to be applied if the **SLA** is not respected) must be specified to ensure that outside suppliers are strongly motivated to abide by the agreement.

Detail Description Paragraph - DETX (591):
[0616] Once the **SLA** is defined, the **resources** required for delivering the service can be specified. Questions to address include the staffing of these **resources** and training to ensure that they are equipped to deliver service as agreed.

Detail Description Paragraph - DETX (622):
[0647] After planning and testing the change to be introduced, it must be implemented. The most common kinds of change in the development environment are the introduction of additional hardware, new releases of **databases,** subroutines and infrastructure, and upgrades to tools. Each change implementation should be viewed as continuous improvement so that any difficulties or inefficiencies are analyzed and resulting improvements are planned and implemented. To be effective over time, this requires that procedures be documented and regularly reviewed and enhanced.

Detail Description Paragraph - DETX (623):
[0648] When the **database** is changed, new versions of test-data must be developed and distributed. When infrastructure components are modified, they may have to be distributed across platforms, and the ripple-effects (for

example, the need for recompilation or code changes in affected components) must be understood and coordinated. Some projects have experimented with incentives to ensure that the infrastructure components do not change too frequently. One such strong incentive is to make the Architecture team responsible for all ripple effects and have them implement all the application level changes that result from an architecture modification.

Detail Description Paragraph - DETX (646):
[0671] Design Application Architecture (see ENACTS MKB **database**)

Detail Description Paragraph - DETX (657):
[0682] Window and **report** design standards

Detail Description Paragraph - DETX (763):
[0788] By the completion of assembly testing, the system should be technically sound, and data flow throughout the system should be correct. Component and assembly testing ensures that all transactions, **database** updates, and conversation flows function accurately. Testing in later stages will concentrate on user requirements and business processes, including work flow.

Detail Description Paragraph - DETX (768):
[0793] Benefits Realization Test--The benefits realization test tests that the business case for the system will be met. The emphasis here is on **measuring** the benefits of the new system, for example: increased productivity, decreased lead times, or lower error rates. If the business case is not testable, the benefits realization test becomes more of a buyer signoff.

Detail Description Paragraph - DETX (769):
[0794] Ideally, benefits realization test occurs prior to complete deployment of the system and utilizes the same environment that was used for the service-level test piece of operational readiness test. Tools are put in place to collect data to prove the business case (e.g., count customer calls). A team of people to **monitor the reports** from the tools and prove that the business case is achieved is still needed. The size of the team depends upon the number of users and the degree to which tools can collect and **report** the data. The benefits realization test tests that the business case for the system will be met. The emphasis here is on **measuring** the benefits of the new system, for example: increased productivity, decreased lead times, or lower error rates. If the business case is not testable, the benefits realization test becomes more of a buyer signoff.

Detail Description Paragraph - DETX (779):
[0804] Service level test--ensures that once the application is rolled out, it provides the level of service to the users as specified in the **Service Level Agreement (SLA)**.

Detail Description Paragraph - DETX (782):
[0807] The operational readiness test is the point in the development process where all the application development, architecture development, and preparation tasks come together. The operational readiness test ensures that the application and architecture can be installed and operated in order to meet the **SLA**.

Detail Description Paragraph - DETX (817):
[0842] On projects with development sites that are geographically distributed, it is usually the case that communication by e-mail alone is not a sufficient substitute for meetings when attempting to coordinate the teams involved. In order to keep all teams updated and moving in the same direction, regular (for example, weekly) conference calls between all parties--chaired by project management--is much more efficient. It is important that these conference calls are closely **monitored,** well prepared, and that the agenda is closely followed. Action points and commitments made during these calls must also be documented. Where issues arise that cannot be resolved using an audio conference (usually because the subject is based on a visual concept), video conferencing may be necessary.

Detail Description Paragraph - DETX (833):
[0858] In a creative environment, it is vitally important that people are able to easily share ideas and information. Teamware provides the ability to capture and share information across a project through the use of common-access, structured **databases**. A good example of teamware is the Knowledge Xchange. Teamware may be used to share many different types of information, for example:

Detail Description Paragraph - DETX (837):
[0862] **Resource** reservation (for example, meeting rooms)

Detail Description Paragraph - DETX (839):
[0864] Status **reports**/meeting minutes

Detail Description Paragraph - DETX (851):
[0876] For example, certain teamware **databases** require continuous maintenance in order to remain relevant. The management of the **database** contents may require significantly more work than either the initial installation of the tools or the technical support for the tools. This effort is frequently underestimated.

Detail Description Paragraph - DETX (855):
[0880] Managing **database** contents

Detail Description Paragraph - DETX (857):
[0882] Managing **resource** usage (for example, disk space)

Detail Description Paragraph - DETX (862):
[0887] Group scheduling tools help to centrally manage the personal schedules of a group of people. This offers the advantage of being able to coordinate events that require the participation of a number of people automatically by checking `group availability` rather than checking with each person individually. These tools may also be used to schedule other **resources** such as meeting rooms and equipment.

Detail Description Paragraph - DETX (922):
[0947] **Reporting**

Detail Description Paragraph - DETX (932):
[0957] **Reporting**

Detail Description Paragraph - DETX (941):
[0966] Mobile code security--protects corporate **resources,** computer files, confidential information, and corporate assets from possible mobile code attack.

Detail Description Paragraph - DETX (958):
[0983] Information Management of the development architecture is provided through an integrated development repository. At this level of integration, tools share a common repository of development objects, design documents, source code, test plans and data. Ideally, the repository would be a single **database** with an all-encompassing information model. Realistically, the repository must be built by integrating the repositories of the different development tools through interfaces. Tool vendors may also build part of the integrated repository by integrating specific products.

Detail Description Paragraph - DETX (990):
[1015] Repository **reports** serve as an audit trail for changes to objects within a repository and can be used to communicate these changes to the entire team. The Repository Management tool should provide this utility. **Reports** for impact analysis are extremely useful in the change control process. As the repository maintains relationships between repository objects, `where-used` and `contains` **report** facilities can be very useful when dealing with change requests.

Detail Description Paragraph - DETX (1033):
[1058] Creating and Changing Data Elements--As soon as data element maintenance becomes structured and is based on formal requests, it is practical to make the requests available to the developers in electronic format. Ideally, the requests should be entered into a **database,** which also contains information on status, comments on the request, and other pertinent information. This **database** can be a useful communication vehicle.

Detail Description Paragraph - DETX (1045):
[1070] m) Analysis, **Reports,** and Queries

Detail Description Paragraph - DETX (1046):
[1071] **Reports** for impact analysis are extremely useful in the change control process. As the repository maintains relationships between repository objects, where-used and contains **reports** are usually provided with the repository. Storing the names of affected repository objects in an area-affected table can be useful when grouping change requests during assignment, or when defining a release. The area-affected table is also a valuable tool that can be used to facilitate migration from development to system test.

Detail Description Paragraph - DETX (1047):
[1072] The ability to easily create various repository **reports** is important to leverage the information in the repository. A scripting language, a simple **report** builder, or a query tool provides this capability. Having a query tool with an intuitive user interface and good **report** formatting features is a necessity on a large project. The query tool can be used to provide standard **reports** for designers and programmers, printed design information for external reviews, and ad hoc requests for the repository administrator.

Detail Description Paragraph - DETX (1081):
[1106] Metrics are an important part of quality management in that they provide a method of **measuring** (for example, sampling, testing, and determining) whether a process or product meets a given criterion. With Metrics, different stakeholders can agree that a product objectively meets an expectation, or that a process has been improved by a measurable amount. Without Metrics, stakeholders can only have subjective opinions that may or may not agree.

Detail Description Paragraph - DETX (1082):
[1107] Measurement tools are used to **measure** process quality and product quality. Process quality may include Metrics such as the time it takes to process a change request. Product quality should be **measured** for all the product expectations the project has set. This measurement process is the inspection part of quality management.

Detail Description Paragraph - DETX (1093):
[1118] Program and Project Management tools assist the management teams in their daily work. These tools, typically packaged as integrated suites of software, provide the basic functionality required for planning, scheduling, tracking, and **reporting** at both the program and project level.

Detail Description Paragraph - DETX (1095):
[1120] Planning tools are used to assist in program and project planning

including the development of the Program **Resource** Plan, the Work Breakdown Structure (WBS), the Organization Breakdown Structure, Cost Accounting, milestones, and deliverables.

Detail Description Paragraph - DETX (1097):
[1122] Scheduling Tools are used to allocate **resources** against the WBS, to determine the timeline for a specific project, and to schedule the allocation of **resources** at the program level.

Detail Description Paragraph - DETX (1099):
[1124] Project tracking tools enable the project manager to track the actual project status against the original plan and schedule. Integration with the time **reporting** system and techniques such as Estimates to Complete (ETCs) are valuable in tracking project status.

Detail Description Paragraph - DETX (1100):
[1125] **Reporting**

Detail Description Paragraph - DETX (1101):
[1126] **Reporting** Tools are used to summarize status and metrics to program and project management.

Detail Description Paragraph - DETX (1125):
[1150] An engagement team must determine whether to purchase a Configuration Management tool or build one. The build decision should consider the cost of designing and developing the functions required by the engagement team. Additionally, the project must consider the **resources** and development time required to build the tool and when the tool is needed in the application development schedule.

Detail Description Paragraph - DETX (1136):
[1161] e) Does the tool provide capabilities for exception **reports**?

Detail Description Paragraph - DETX (1137):
[1162] If for some reason a repository component is not at the correct promotion level, the tool should be able to **report** on this when required.

Detail Description Paragraph - DETX (1144):
[1169] The tools should automate the storage and retrieval of all dependent software components indicated by an impact analysis **report**.

Detail Description Paragraph - DETX (1150):
[1175] Version Control tools allow systematic storage of information about who makes changes in what order so that the evolution of the system can be

tracked. The tools usually provide a facility to **report** on differences in versions so the version that existed when a critical change was made can be identified and recreated or retrieved. The tools can also provide a means of documenting why decisions are made during the evolution of the system. These decisions would have been made based on the version of the documentation for the system that existed at that time. Version Control tools allow the state of the system at a particular time to be recorded. Hence improved auditability for decisions can be achieved.

Detail Description Paragraph - DETX (1175):
[1200] Flexible, customizable sorting and **reporting** to ensure that a change is handled in a timely manner

Detail Description Paragraph - DETX (1199):
[1224] Change requests may occur as a consequence of changing requirements, or as a result of nonconformities (or defects) in the system. The tool should be able to classify change requests into categories such as incidents, faults, or enhancements. The tool should also have the ability to update these categories if required. Classification of different change requests in several different ways such as area affected, priority, estimated cost or authorization is important to ensure correct scheduling of the implementation of changes. Flexible, customized sorting and **reporting** based on this classification is required to ensure that change is handled in a timely manner.

Detail Description Paragraph - DETX (1202):
[1227] If an impact analysis tool cannot be found that supports the entire environment, it is critical to develop procedures or utilities that will **report** on where items are used. The first step is to identify the items to be searched, and to build procedures around searching them (for example, **databases,** files, workspaces, programs, screens/forms, **reports**). It is also important to identify who will be responsible for the impact analysis (DBA, analysts, programmers, team leaders, and so on) to avoid this work falling between the cracks.

Detail Description Paragraph - DETX (1206):
[1231] The tool should allocate change requests to different releases based on priority and **resource** availability. It should also provide a means of attaching a deadline to a change request.

Detail Description Paragraph - DETX (1210):
[1235] The tool should provide a capability to generate exception **reports** that highlight issues such as change requests that are in danger of not meeting the release to which it was allocated. f) What is the prediction for volume of change requests for the project?

Detail Description Paragraph - DETX (1216):
[1241] i) What **reporting** capabilities are needed on the project?

Detail Description Paragraph - DETX (1217):
[1242] Some Change Control tools can **report** on status of change requests at the individual, team, and project level. Such **reports** can provide information about work done to date and Estimate to Complete (ETC) values.

Detail Description Paragraph - DETX (1227):
[1252] Migration Control tools control multiple versions of source code, data, and other items as they are changed, tested, and moved from one development environment into another, for example, from development to test and from test to production. Data migration control tools manage multiple versions of the **database** and its data to ensure that accurate data and structure are maintained in the environment, and to ensure that versions of application code and **database** are deployed consistently. Types of data that would be migrated include base codes data and converted data. Other Migration Control tools manage other types of objects to ensure that complete versions of all components reside in the production environment (for example, test definitions and scripts).

Detail Description Paragraph - DETX (1241):
[1266] The Migration Control tool should be able to manage and control the migration of all the components (for example, source code, **database** access, make files, run-time data, environment variables, code libraries, code tables, third-party software, and so forth) which make up the object to be migrated. The complexity of the Netcentric world with so many integrated vendor solutions dramatically increases the number and variations of object types.

Detail Description Paragraph - DETX (1251):
[1276] **Monitoring** functionalities, in order to **measure** progress towards delivery goals

Detail Description Paragraph - DETX (1268):
[1293] Tools to support the creation, management, and **reporting of Service Level Agreements (SLAs)** and Operations Level Agreements (OLAs)

Detail Description Paragraph - DETX (1273):
[1298] A comprehensive development environment rapidly becomes sufficiently complex that the startup and shutdown of the environment must be managed carefully, and preferably automated. This is key to ensuring the integrity of the environment. Startup may involve the carefully sequenced initialization of networking software, **databases,** web servers and more. Similarly, shutdown involves saving configuration changes as needed and gracefully taking down running software in the correct sequence.

Detail Description Paragraph - DETX (1290):
[1315] Performance **Monitoring**

Detail Description Paragraph - DETX (1291):

[1316] Performance **Monitoring** tools help ensure that the available **resources** are sufficient to meet the developers' performance requirements. These tools can be used to assess end-to-end performance of both batch processes such as backups, and interactive processes such as repository-based file retrieval.

Detail Description Paragraph - DETX (1304):

[1329] Managing Change tools support the various aspects of identifying and managing change in the development environment. Specific tools are discussed in detail in the MODE Products **Database** on the Knowledge Xchange.

Detail Description Paragraph - DETX (1308):

[1333] Problem Management tools log information about problems detected, classify them, and generate **reports**. This is essential for capturing metrics information.

Detail Description Paragraph - DETX (1325):

[1350] Stage containment is an approach to identify problems in the system before they pass to the next stage. It is a **measure** that helps build quality into the system. The goal of stage containment is to minimize the number of errors being passed to the next stage. For the purpose of stage containment, problems are sorted into categories. Errors are defined as problems found in the stage where they were created. Defects are problems found in a stage successive to the stage where they were created. Faults are problems found in production. The longer a defect remains undiscovered, the more difficult and expensive it will be to correct. Because each stage relies on the decisions made during the creation of the specification in the previous stage, detecting an error in a stage after it was made may invalidate some or all of the work done between the time the issue was created and the time it was discovered.

Detail Description Paragraph - DETX (1341):

[1366] Problem Management tools log error information, generate error **reports** (such as System Investigation **Reports** or SIRs), classify problems, and record information on the source of the error. Problem Management tools are essential for the capture of stage containment metric information.

Detail Description Paragraph - DETX (1357):

[1382] As systems are often built on top of legacy **databases,** some data modeling tools allow generation of an object model from the legacy **database** data model (DDL). By understanding the E-R diagram represented by the **database,** it is easier to create an efficient persistence framework which isolates business components from a direct access to relational **databases**. Caution is required, however, as the resulting model is at best only partial, as an object model has dynamic aspects to it as well as static relationships, and may not correctly reflect the analysis performed in the problem domain.

Detail Description Paragraph - DETX (1358):
[1383] When a component or object-based approach is used, data modeling is not performed. Rather, the object model contains both the data and the behavior associated with an object. In most systems relational **databases** are used and the object model must be mapped to the data model. Standard mechanisms for mapping objects exist. Tools such as Persistence (Persistence Corp.) and DBTools (Rogue Wave) can generate the code necessary to map objects to a **database**.

Detail Description Paragraph - DETX (1363):
[1388] Data modeling tools help to graphically develop the logical and physical data requirements for an application. These tools depict logical constructs such as entities, attributes, and relationships between entities, along with physical constructs such as **database** definitions and table indices.

Detail Description Paragraph - DETX (1367):
[1392] Data modeling tools promote consistency in application development by defining standard names and attribute characteristics for the application data. Application developers then use the standard entity and attribute definitions across various application development initiatives. This results in a consistent definition and usage of data. For example, all applications that require customer number will use the standard name and attribute length defined in the data model. **Database** administrators will also use the data model to generate physical **database** definitions that are consistent with the application under development. Thus, the data model acts as a single source for data definition.

Detail Description Paragraph - DETX (1372):
[1397] Is a relational **database** being used to store persistent objects?

Detail Description Paragraph - DETX (1373):
[1398] Fully normalized data models are a different view of the corresponding object models. On the one hand, the data model does not show behaviors (methods). On the other hand it does show resolving entities that are normally modeled as container objects and may be internal to an object. A data modeling tool is useful for showing how the persistent objects map to the relational **database**.

Detail Description Paragraph - DETX (1378):
[1403] g) Is **database** design going to be performed?

Detail Description Paragraph - DETX (1379):
[1404] The finalized data model is used as a basis for the logical **database** design. The logical **database** design converts the finalized Project Data Model to one of four basic structures, according to which DBMS is used:

Detail Description Paragraph - DETX (1384):

[1409] Although entity-relationship diagrams are independent of specific DBMSs or access methods, a logical **database** design is not. This design is highly dependent on the platform components and may need to be repeated for each location type and platform type. This process is simplified if a data model is used.

Detail Description Paragraph - DETX (1386):

[1411] Data modeling tools allow documentation of the data in so far as it appears in the data model (and ultimately in the **database**). However, there is usually a significant number of other data definitions which will never appear in the **database**, and whose definition is different to the data model attributes. For example, most systems have interfaces to external systems, and inherit a legacy of interface files whose data definitions may differ to those on the data model, but which do logically correspond to fields on the model. These data definitions must also be documented and stored but are effectively outside the data model. The data modeling component should be used to implement procedures to address all the data definitions that affect the system.

Detail Description Paragraph - DETX (1389):

[1414] The features required in the data modeling tool will depend on the intended use of the tool. If the tool is to be used to develop logical data models, it should support logical constructs such as entity definition, attribute definition, subtyping, and supertyping. If the tool is to be used for physical data design, it should support the physical constructs required for the targeted RDBMs, such as transforming a logical model into a physical model, **database** definition, index definition, and DDL generation.

Detail Description Paragraph - DETX (1393):

[1418] It is important to consider the various utilities available with the data modeling tools. Two such utilities include impact analysis and **reporting**. Impact analysis capabilities allow the user to understand the impact of a change to the data model. Impact analysis functionality is one of the key tools used by engagement teams to assist with change management and change control activities. Some products will also include **report** generators which are useful for generating data and attribute definition **reports** as well as ad hoc **reports**.

Detail Description Paragraph - DETX (1397):

[1422] Data modeling tools commonly integrate with the repository and with system building tools such as window painters and Application Logic Design tools. If the tool does not provide seamless integration with other components of the development environment, the engagement team can build bridges between components, or develop manual procedures in order to share information. It is important to consider how the data modeling tool integrates with the design repository. It is important to maintain a cross-reference of the attributes on the model, with the definition of data elements in the design repository. Such

data element definitions will also address non-**database** data definitions (e.g. external i/face files).

Detail Description Paragraph - DETX (1401):
[1426] g) Should the data modeling tool provide **database** design facilities?

Detail Description Paragraph - DETX (1403):
[1428] Most data modeling tools allow you to develop the **database** design at the same time. This has the advantage of keeping costs down as two separate tools need not be purchased, and of ensuring consistency by providing a direct interface between the two phases.

Detail Description Paragraph - DETX (1407):
[1432] The information management component may provide the security needed in a multi-designer environment. If this is not the case then a multi-designer data modeling tool should be used. The tool may provide a central dictionary which allows design data to be shared between several designers and includes security checks to **monitor** any conflicts in overlapping access rights between designers.

Detail Description Paragraph - DETX (1490):
[1515] Dedicated performance modeling tools should be considered for any project that involves high transaction volumes or a complex architecture with several platforms. Performance is critical for such systems and a performance model is required in order to predict and **optimize** that performance.

Detail Description Paragraph - DETX (1504):
[1529] Specific modeling tools can provide advantages such as cross referencing (for example, are all the methods used in the Interaction diagrams described in the class definitions?), automatic propagation of changes to other diagrams, generation of **reports,** and generation of skeleton code. However, some tools have problems with:

Detail Description Paragraph - DETX (1530):
[1555] Prototyping can address this problem by simulating key user interface components, thus enabling the development team to **measure** the usability of the proposed system at a very early stage. The most important quality of a prototyping tool is its development speed. If prototyping can be performed in hours or days rather than weeks or months, it becomes possible to perform more iterations, which explore different options. This may lead to a much better system, given that the user's perception matures with each iteration. This, in turn, improves the quality of user input.

Detail Description Paragraph - DETX (1553):
[1578] If the system is to be used by dedicated people where the **measure** of productivity is solely the number of transactions they can get through per

second, then user interface prototyping tools are important. Prototyping tools provide a means of getting to the easiest and most efficient interface. Prototyping tools facilitate selection between alternative styles of interaction and provide a means of addressing performance issues.

Detail Description Paragraph - DETX (1623):
[1648] **Database** Design

Detail Description Paragraph - DETX (1624):
[1649] **Database** design tools provide a graphical depiction of the **database** design for the system. They enable the developer to illustrate the tables, file structures, etc., that will be physically implemented from the logical data requirements. The tools also represent data elements, indexing, and foreign keys.

Detail Description Paragraph - DETX (1625):
[1650] Many data design tools integrate data modeling, **database** design, and **database** construction. An integrated tool will typically generate the first-cut **database** design from the data model, and will generate the **database** definition from the **database** design.

Detail Description Paragraph - DETX (1626):
[1651] With an object-based or component-based solution the data modeling task changes. In most cases, relational **databases** are still used, even where there are no dependencies on legacy systems. As there is an `impedance mis-match` between an object model and a data model, a mapping activity must be undertaken. There are standard mechanisms for doing this.

Detail Description Paragraph - DETX (1628):
[1653] There can be performance problems with objects mapped to a relational **database**. In a worst case scenario, an object can be spread across many tables, with a single select/insert for each table, and as each object is loaded one by one, the performance becomes very poor. Some tools provide lazy initialization (only loading the parts as they are needed) and caching (minimizing DB hits).

Detail Description Paragraph - DETX (1629):
[1654] The current trend seems to be for object-relational **databases,** with vendors such as Oracle adding object features to their core products. Although the support provided at the moment is limited, it is likely that in future versions Java or C++ classes will be able to interface directly.

Detail Description Paragraph - DETX (1632):
[1657] **Database** design tools are important where design ideas must be communicated to the development team. Where the development team exceeds ten people, this design must be formalized. **Database** design tools provide a

graphic depiction of the **database** design for a system, whilst at the same time enabling the developer to illustrate tables and other structures that will be implemented physically.

Detail Description Paragraph - DETX (1634):
[1659] **Database** design tools become especially important if performance is critical, since **database** design contributes substantially to the overall performance of the system. **Database** design tools provide quantifiable performance data which is a crucial component of the overall performance model.

Detail Description Paragraph - DETX (1635):
[1660] **Database** Design tools also provide a means to model I/O on devices such as hard disks, optical drives, and tapes etc. This information can be used in a performance model.

Detail Description Paragraph - DETX (1637):
[1662] The **database** design component is important in the case where multiple teams are working on different functional domains, since they often model different parts of the **database** separately and then incorporate these models at the end into one large **database** model. **Database** design tools can be used to enforce consistency of the different **database** designs.

Detail Description Paragraph - DETX (1638):
[1663] d) Does the **database** include a very large number of tables and elements?

Detail Description Paragraph - DETX (1639):
[1664] Navigation through a large number of tables is complicated and can be simplified significantly if dedicated **database** design tools are used.

Detail Description Paragraph - DETX (1641):
[1666] Different teams or users may have different requirements which conflict. These requirements may have to be rationally traded-off against each other. Where these requirements are performance related, the trade-off can only be rationalized on the basis of a good **database** model.

Detail Description Paragraph - DETX (1647):
[1672] b) Should the **database** design tools support **database** construction?

Detail Description Paragraph - DETX (1648):
[1673] Many **database** design tools allow for **database** construction. Such tools may help translate a logical **database** design into a physical design, or they may generate Data Definition Language (DDL) code or Data Manipulation Language (DML) code. The advantage of using a tool that provides this facility is that it simplifies the transfer of design information into a physical

representation and can be used to ensure consistency from design into construction of the **database**.

Detail Description Paragraph - DETX (1650):
[1675] Presentation design tools provide a graphical depiction of the presentation layer of the application, such as windows, dialogs, pages, navigation and **reports**. Tools in this category include window editors, **report** editors, and dialog flow (navigation) editors. Window editors enable the developer to design the windows for the application using standard GUI components. **Report** editors enable the developer to design the **report** layout interactively, placing literals and application data on the layout without specifying implementation details such as page breaks. The majority of these tools generate the associated application code required to display these components in the target system.

Detail Description Paragraph - DETX (1682):
[1707] It is important to determine how well the product integrates with other design and development tools, presentation services (graphics, multi-media, etc.), data access services (**databases and database** API libraries), distribution services (distributed TP **monitor**), transmission services (SNA, HLLAPI, etc.), data dictionary, desktop applications, and programming languages for call-out/call-in. Additional consideration should be given to add-on and third-party products/enhancements such as specialized widgets, **report** writers and case tools.

Detail Description Paragraph - DETX (1687):
[1712] g) What **databases** are supported?

Detail Description Paragraph - DETX (1688):
[1713] h) What protocols are used to communicate with the **database**?

Detail Description Paragraph - DETX (1689):
[1714] Important considerations include the supported **databases** and protocols used to communicate with the **databases**. The tool must support the selected **database**. Additionally, if **database** selection may change, it is important that the tool have the ability to support other **databases** with minimal impact on the application development. Native **database** interfaces tend to have better performance than open standards such as ODBC.

Detail Description Paragraph - DETX (1702):
[1727] Special characters, differences in field lengths, and differences in number formats are some of the things that contribute to the complexity of a multi-language application. Window and **report** design are among the areas affected by differences in the language used for presentation.

Detail Description Paragraph - DETX (1706):

[1731] The presentation design tools should be tightly integrated with the system components stored in the repository, such as windows, **reports,** screens, and other more abstract models to ensure consistency.

Detail Description Paragraph - DETX (1747):

[1772] Developers use interactive navigation tools to identify requirements for a new system from the functionality and design of a legacy system. These tools enable the developer to interactively and graphically navigate the legacy system, determining the system's characteristics such as system structure, module flow, flow control, calling patterns, complexity, and data and variable usage. An alternate form of presentation is through **reports**. These provide cross-reference listings or graphical representations of control or data flows.

Detail Description Paragraph - DETX (1749):

[1774] Graphical representation tools are used to display important system information in a form, which is easier to assimilate. These tools may, for example, produce structure charts, **database** schema diagrams, and data layouts. They can also print matrices that indicate relationships between modules and files or between jobs and programs.

Detail Description Paragraph - DETX (1751):

[1776] An extraction tool, in conjunction with a repository population tool, enables the developer to reuse selected portions of a legacy system. The extraction tool can typically read and extract information from source code, screens, **reports, and the database**. The most common information extracted from a legacy system, however, is the data: record/table structure, indexes, and data element definitions.

Detail Description Paragraph - DETX (1752):

[1777] In component-based architectures, as systems are often built on top of legacy **databases,** some extraction tools allow generation of an object model from the legacy **database** data model (DDL). By understanding the E-R diagram represented by the **database,** it is easier to create an efficient persistence framework which isolates business components from a direct access to relational **databases**. Caution is required, however, as the resulting model is at best only partial, as an object model has dynamic aspects to it as well as static relationships, and may not correctly reflect the analysis performed in the problem domain.

Detail Description Paragraph - DETX (1760):

[1785] Packaged components are generally third party components that provide ready-made business logic that is customizable and reusable. These can range from simple components offering limited functionality (for example, worksheet or charting GUI components), to components that handle a significant portion of the application architecture (for example, data access components and firewalls). The advantage of using such components is that they have already been coded, tested, **optimized,** and documented.

Detail Description Paragraph - DETX (1770):
[1795] Components are often developed with a preferred platform in mind. Components **optimized** for one platform may have severe performance problems on others. If performance is a factor (and it nearly always is) ensure that components are designed specifically for the platform of the target system.

Detail Description Paragraph - DETX (1779):
[1804] Construction tools are used to program or build the application: client and server source code, windows, **reports, and database**. Along with the onset of Visual Programming, the more traditional form of construction tools have been superceded by Integrated Development Environments (IDEs) which take all the basic components required for construction, and integrate them into a single system. Although IDEs are now the preferred tools for most construction, the components that make up these tools remain the same--Source Code Editor, Compiler/Linker/Interpreter, Generation Tools and Debugging Tools.

Detail Description Paragraph - DETX (1825):
[1850] Symbolic source code enables easier identification of where errors occur. Preferably, the debugger should be flexible enough to work with any combination of compiled modules and source modules. In addition, the debugger should be able to handle calls to the **database** and to other modules.

Detail Description Paragraph - DETX (1856):
[1881] Generators are used to leverage the powers of code reuse and code regeneration. The ability to reuse code reduces both the time and **resources** required on a project. Code regeneration eases maintenance issues by propagating changes throughout multiple sections of code.

Detail Description Paragraph - DETX (1859):
[1884] The code/applications generated by the tools vary in performance. **Optimized** code usually results in faster run times. It is important to identify the high priority components that will benefit most from the tool.

Detail Description Paragraph - DETX (1861):
[1886] The decision to custom build or to buy available case tools must be determined by the development team. Most generators are usually custom built because often the technical environment and architecture have custom components that cannot be handled by a package generator. Associated with custom building are the issues of added cost and development time, but performance can be closely **monitored** and changes performed on the spot.

Detail Description Paragraph - DETX (1866):
[1891] Code Analysis--Code analysis provides the objective information and metrics needed to **monitor** and improve code quality and maintenance (e.g. static analyzer, documentor, auditor).

Detail Description Paragraph - DETX (1872):
[1897] Code and Object libraries provide the developer with ready-made components (such as GUI components or simple utilities), which may be integrated into architecture or application code. The advantage of using such components is that they have already been coded, tested, **optimized,** and documented.

Detail Description Paragraph - DETX (1910):
[1935] In some cases, the mentioned entities and relationships cannot be managed within the repository, and may have to be modeled outside the repository (for example, in a teamware **database**). In this case, the link between the repository and the external tools must be provided by a judiciously chosen set of procedures and custom integration tools.

Detail Description Paragraph - DETX (1913):
[1938] Year 2000 Testing Contacts and KX **Resources**

Detail Description Paragraph - DETX (1938):
[1963] Test Data Manipulation tools are used to create original test data and, sometimes, to modify existing test data. Such modifications may be needed to process a change in the **database** schema and to correct intermediate results in order to complete a test cycle. Some test data manipulation tools generate test data very effectively.

Detail Description Paragraph - DETX (1964):
[1989] f) What is the importance of a test **database**?

Detail Description Paragraph - DETX (1969):
[1994] In order to plan detailed script steps and expected results, it is necessary to know the test data. A large portion of the test data will typically be contained in test **databases**. These **databases** are called baseline **databases,** and are critical for a repeatable test model to exist. Baseline **databases** can be developed automatically (through execution of online activity in the system), manually (through test data manipulation tools), extracted from production **databases,** and so on. Once the baseline **databases** are selected and created, the repeatable test model can be developed. As the test model is based upon these **databases,** the impact on the test model of any changes to the baseline **databases** must be analyzed.

Detail Description Paragraph - DETX (1971):
[1996] If the application does not change, repeating the tests yields the same results every time, given the same baseline **databases**. To remain repeatable, a test model must be maintained to reflect changes made to the application (fixes, isolated enhancements, new releases, and so on).

Detail Description Paragraph - DETX (1974):
[1999] Automating the execution of a non-repeatable test model is a waste of **resources,** as the test tool will not be able to re-execute the tests automatically or perform full regression tests with little effort. Little or no benefits will be achieved from automation.

Detail Description Paragraph - DETX (1983):
[2008] A **reporting** option is planned to produce metrics and management type **reports**.

Detail Description Paragraph - DETX (2043):
[2068] Performance Management tools support application performance testing. Owing to the large number of components in modern systems, performance modeling can be a complex task and requires tools to effectively manage the process. These tools **monitor** the real-time execution and performance of software. They help to maximize transactions and response time to the end user. They are also useful in identifying potential bottlenecks or processing anomalies.

Detail Description Paragraph - DETX (2044):
[2069] In the case of Internet-based applications, as the Internet is not a controlled environment, performance management tools can only **measure** performance within the domain of the controlled environment (up to the Internet Service Provider). However, in the case of intranet-based systems, where the environment is controlled from end-to-end, Performance Management may be performed across the entire system.

Detail Description Paragraph - DETX (2051):
[2076] Test Result Comparison tools are utilities used to compare expected and actual results. These tools outline the differences between actual and expected results by comparing files and **databases**. Most of these tools offer functionality such as byte-by-byte comparison of files and the ability to mask certain fields such as date and time.

Detail Description Paragraph - DETX (2059):
[2084] The following **databases** provide information on the Operations Architecture and list requirements and current tools solutions for the managing of the various Operations Architecture areas. All areas of the Operations Architecture have the appropriate MODE sub-functions listed, along with requirements for management solutions and current tools that assist and automate management solutions.

Detail Description Paragraph - DETX (2088):
[2113] Will existing data/**databases** be used, or will data be built from scratch?

Detail Description Paragraph - DETX (2097):

[2122] Systems-based tools (e.g., for **monitoring** or control purposes) will clearly be platform dependent. Functional tools (e.g., to support Incident Management or Change Control), however, can run independently from the systems tools and may only need to run on a limited number of systems.

Detail Description Paragraph - DETX (2102):

[2127] Understanding the anticipated volumes will provide key input to sizing the system. Predicted business volumes stated in the **SLA** should be used to help determine the appropriate sizes for machines, **databases,** telecommunications lines, etc. Alternatively, experience from previous engagements can provide useful input.

Detail Description Paragraph - DETX (2110):

[2135] The presentation component provides the interface between the manager(s) of the system and management data generated by the system. Data can be manipulated for various forms of output. By integrating the operational architecture it is possible to reduce the number of front-end interfaces required. Commonly, the presentation component uses a GUI front-end interface. This component is also responsible for real-time and historical **report** generation.

Detail Description Paragraph - DETX (2114):

[2139] As with End User Services in the centralized model, the Help Desk is the single point of contact for all end users. This unit has end-to-end accountability for all user incidents and problems regardless of whether or not it has the **resources** to fix them (i.e., it must contact the necessary technical **resources** in either IS organizations to ensure the incidents and problems get resolved).

Detail Description Paragraph - DETX (2130):

[2155] **Monitoring** (1316)

Detail Description Paragraph - DETX (2135):

[2160] Management applications are those tools which are used to manage the system. Most of the MODE functions tie directly into this component. The management applications component ties in directly with the integration platform component as the management applications tools must comply with the standards set by the integration platform. For example, if the integration platform is HP OpenView, then the management applications must be HP OpenView software (API, SNMPx) or hardware (card) compliant. Management applications receive data from the event/data generation, event processing, and repositories components and then send data to the presentation or repositories components. Management applications tools include capacity planning tools, performance management tools, license management tools, remote management tools, systems **monitoring** tools, scheduling tools, help desk tools, etc. Some Enterprise Management tools even poll the event/data generators for information but these options may impact network performance. Web Server management is been introduced as part of the management operations framework. As Corporate

Internets and Extranets implement Web based software products to sell and advertise business services, corresponding administrative, security, event notification and performance requirements must be performed similarly for the companies web based system. The critical path issues for Web based server software is typically security and performance based levels of service.

Detail Description Paragraph - DETX (2137):
[2162] As with End User Services in the centralized model, the Help Desk is the single point of contact for all end users. This unit has end-to-end accountability for all user incidents and problems regardless of whether or not it has the **resources** to fix them (i.e., it must contact the necessary technical **resources** in either IS organizations to ensure the incidents and problems get resolved).

Detail Description Paragraph - DETX (2147):
[2172] Incidents and requests should be closed with a date and time stamp to help trend analysis and service level **reporting**.

Detail Description Paragraph - DETX (2149):
[2174] Problems can be logged both as a result of one or more incidents, or through proactive **monitoring** of the system, before any incidents have been logged.

Detail Description Paragraph - DETX (2151):
[2176] If the Incident, Request and Problem management functions are to be centralized, these functions need to be able to control and **monitor** incidents and problems, but other functions should be able to gain access to input detailed technical information or progress updates. If Incident and Request management is distributed, it is recommended that remote locations are given access to the central system, rather than operating local systems. (Some problem areas are local sites operating on different time zones and standardizing escalation procedures from local sites.)

Detail Description Paragraph - DETX (2153):
[2178] Event/alert based automatic logging of incidents to provide proactive management of incidents and problems by informing Incident management of issues before the user logs a call. This facility is conceptually desirable, but is only likely to be available if the Incident management functionality is part of the **monitoring** tool. The costs of building hooks between tools and applications are likely to prove prohibitive. In medium or large environments, this facility is extremely desirable, and must be built into the requirements.

Detail Description Paragraph - DETX (2187):
[2212] In the event of a significant system failure, Disaster Recovery processes will be invoked to re-route the system **resources** to a secondary, stable configuration until the primary **resources** can be restored. Within a distributed environment, disaster recovery must account for differing levels of

disaster whether at a central or distributed site(s).

Detail Description Paragraph - DETX (2190):
[2215] The way in which a disaster is defined will be dependent upon which **resources** are critical to the business. For example, a data center failure may be critical for one client whereas a server failure for another is more critical.

Detail Description Paragraph - DETX (2192):
[2217] This will be defined in detail within the **SLA,** but high level service recovery targets must be understood, so that high level recovery plans can, in turn, be produced.

Detail Description Paragraph - DETX (2199):
[2224] Hardware Maintenance maintains all of the components within a distributed system to protect the investment of the organization. Generally agreed upon in the **SLAs,** maintenance contracts are carried out, **monitored** and recorded for each asset as appropriate.

Detail Description Paragraph - DETX (2203):
[2228] Billing & Accounting also makes payments to service providers for services and equipment provided in accordance with agreed upon **SLAs**. As part of this payment process Billing & Accounting reconciles bills from service providers against **monitored** costs and **SLA**/OLA violations.

Detail Description Paragraph - DETX (2206):
[2231] Capacity Modeling & Planning ensures that adequate **resources** will be in place to meet the **SLA** requirements, keeping in mind operational requirements which may require additional capacity. **Resources** can include such things as physical facilities, computers, memory/disk space, communications lines and personnel. Through this component, changes to the existing environment will be determined, modeled and planned according to the necessary requirements.

Detail Description Paragraph - DETX (2225):
[2250] The tool should provide control dependencies to schedule workloads such as: Task/job sequence enforcement, external/internal event driven. Graphically displays work flow from the scheduling criteria and includes such information as task/job name, task description, average run time and **resource** requirements. Allow clients to define user schedules that can be based on predecessor events in the production environment. **Reporting** capabilities for forecasting, simulation and analyzing scheduled workload. **Monitoring** capability of past, present and future workloads as well as tracking of current workload termination notification of normal or abnormal completion.

Detail Description Paragraph - DETX (2231):
[2256] Communication with Performance management component to forecast

resource requirements, such as near line storage, DASD space, and etc.

Detail Description Paragraph - DETX (2237):
[2262] Print Management **monitors** all of the printing done across a distributed environment and is responsible for managing the printers and printing at both central and remote locations. The purpose of a print architecture is to make formats applications independent, so that the only thing applications need to do is obtain the data.

Detail Description Paragraph - DETX (2240):
[2265] It makes it easy to develop and maintain **report**

Detail Description Paragraph - DETX (2242):
[2267] **Reports** arrive to the addressee more quickly

Detail Description Paragraph - DETX (2243):
[2268] It is possible to sign **reports** electronically

Detail Description Paragraph - DETX (2251):
[2276] If spooling is available, printing can be handled as a background task, freeing up system **resources** for use on-line.

Detail Description Paragraph - DETX (2255):
[2280] Large print jobs may utilize system **resources** considerably (e.g., WAN, LAN, printer), and may tie up the printing queue for other individuals. This type of printing should be performed in off-hours or delayed to avoid contention for the printer during business hours.

Detail Description Paragraph - DETX (2260):
[2285] Controls **report** production and distribution form the moment the **report** is created to the time the printed **report** is dropped in the end-use s mailbox (electronic, paper, microfiche, etc.)

Detail Description Paragraph - DETX (2264):
[2289] Provides for the archival of **reports** in a compressed format first on disk, for a user specified time and then to tape of optical.

Detail Description Paragraph - DETX (2265):
[2290] Process **reports** in due-out-sequence.

Detail Description Paragraph - DETX (2266):
[2291] Automatic **report** balancing and archives the balancing **reports** for easy auditor review.

Detail Description Paragraph - DETX (2268):
[2293] Provide **report** reprint capability, avoid reruns in lost **report** situations.

Detail Description Paragraph - DETX (2269):
[2294] Provide centralized management of **report** setup and delivery information

Detail Description Paragraph - DETX (2271):
[2296] Interfaces with the performance **monitoring** to identify bottlenecks in the distribution process

Detail Description Paragraph - DETX (2274):
[2299] Communicates with the recovery management facility to delete **reports** that will be recreated.

Detail Description Paragraph - DETX (2275):
[2300] Communicates **report** volumes to the **resource** consumption management facility.

Detail Description Paragraph - DETX (2277):
[2302] Support multiple printer types as well as **report** delivery across them. This includes printer format translation (PCL, Postscript, etc.) and code translation.

Detail Description Paragraph - DETX (2281):
[2306] File Transfer and Control initiates and **monitors** files being transferred throughout the system as part of the business processing (e.g., nightly batch runs). File transfers may occur between any two or more devises within the system.

Detail Description Paragraph - DETX (2283):
[2308] System Startup and Shutdown performs the activities required for the startup or shutdown of the entire system (e.g., hardware, applications), or portions of the system depending upon the identified requirements. Within a distributed environment, the system includes both centralized and remote **resources**.

Detail Description Paragraph - DETX (2290):
[2315] Based upon the technical requirements of the system (e.g., **databases** should be started before applications) as well as defined service levels (e.g., one particular application is critical and must be started first), the order of startup/shutdown will be determined.

Detail Description Paragraph - DETX (2293):
[2318] Analysis of the system and other **resources** need to be addressed?

Detail Description Paragraph - DETX (2294):
[2319] The state of an application, the system or a specific **resource** must be known at all times. Common activities performed as part of Startup/Shutdown include:

Detail Description Paragraph - DETX (2304):
[2329] Mass Storage Management involves those activities related to the handling of various types of centralized and distributed storage media including the **monitoring** and controlling of storage **resources** and their usage.

Detail Description Paragraph - DETX (2310):
[2335] Will **databases** be distributed or centralized?

Detail Description Paragraph - DETX (2311):
[2336] Storage management for centralized **databases** will clearly be simpler then for distributed **databases** were a global view becomes more difficult to obtain, and where data consistency becomes more of an issue.

Detail Description Paragraph - DETX (2327):
[2352] Does the tool provide support for the **databases** selected for the distributed environment?

Detail Description Paragraph - DETX (2328):
[2353] Additional facilities may be required, even although **databases** typically have built-in utilities or tools to perform these function and do not generally require a separate tool.

Detail Description Paragraph - DETX (2332):
[2357] Interface with the Capacity/**Resource** manager to create a definable **resource** forecast.

Detail Description Paragraph - DETX (2342):
[2367] Because **databases** can be located throughout the distributed environment, care must be taken to ensure that data integrity is maintained. This may mean storing the master copy of data centrally, or synchronizing the commits of updates of the information appropriately.

Detail Description Paragraph - DETX (2355):
[2380] Designed along the lines requester-server model; more specifically

the tool runs on the server machine and acts as a shared **resource** for data access, integrity, security recovery, etc.

Detail Description Paragraph - DETX (2362):
[2387] **Database** backup/restore is inherently more complex than backup of standard files. It is important to ensure that all relationships are resurrected after restoring **database** files. (Integrated with the functionality of the DBMS)

Detail Description Paragraph - DETX (2367):
[2392] Pint in time recovery of **database and database** components must be supported.

Detail Description Paragraph - DETX (2371):
[2396] Does the tool add color to MODE architecture model through performance **measures**?

Detail Description Paragraph - DETX (2376):
[2401] Which files and **databases** will be archived?

Detail Description Paragraph - DETX (2377):
[2402] Some files and **databases** need to be stored on fast devices so users can access them quickly. In addition, certain files may need to be maintained for either historic or government/regulatory reasons.

Detail Description Paragraph - DETX (2383):
[2408] Because **databases** can be located throughout the distributed environment, care must be taken to ensure that data integrity is maintained. This may mean storing the master copy of data centrally, or synchronizing the commits or updated of the information appropriately.

Detail Description Paragraph - DETX (2394):
[2419] Configuration settings can be retrieved from different sources. The release and the rollout schedule will contain a detailed description of equipment and its configuration and can therefore be used as input. Alternatively, the asset inventory system can be updated in advance and then used as an active **database** to drive the configuring process.

Detail Description Paragraph - DETX (2405):
[2430] The software and data distribution mechanism itself updates either the software, data, or configuration information on a machine(s), **reports** the relative success/failure of the distribution and updates the asset information for the sites/machine(s) affected by the distribution.

Detail Description Paragraph - DETX (2410):
[2435] The existing skills must be assessed and a forward-thinking training direction must be defined. The training plan will likely emphasize newer technologies and different methods of training with the underlying goal of providing the appropriate level of service as required by the **SLAs**.

Detail Description Paragraph - DETX (2418):
[2443] Ability to generate distribution candidate lists from asset/inventory management **database**.

Detail Description Paragraph - DETX (2419):
[2444] Logging of status/failures to centralized system **monitoring** facility.

Detail Description Paragraph - DETX (2434):
[2459] Specialized functionality to support operation across the wide-area network environment including: parallel distribution and data compression. In addition, support of platform specific functions and capabilities due to awareness of platform specific information resident in the asset/inventory **database**.

Detail Description Paragraph - DETX (2445):
[2470] Due to the number of components, users may be required to have multiple ID(s) and passwords unless the system is designed to allow a user to access all of the required **resources** through a single logon. As most products on the market typically allow access to only a subset of **resources,** single logons with multiple ID and password coordination may be difficult to achieve. Issues such as periodic required password changes can be difficult to overcome while maintaining adequate security.

Detail Description Paragraph - DETX (2448):
[2473] Protects all computer **resources,** facilities and data from accidental or intentional destruction, modification, disclosure and/or misuse.

Detail Description Paragraph - DETX (2451):
[2476] Maintains a security log and user profile of what was accessed when, from a computer **resource,** facility and data view point.

Detail Description Paragraph - DETX (2452):
[2477] Security Administration ability to **monitor** the activity of a user of **resource**.

Detail Description Paragraph - DETX (2454):
[2479] Access authority for **database** objects (data-sets) as they appear outside the DBMS must be controlled.

Detail Description Paragraph - DETX (2455):
[2480] **Database** authorities must be manageable at a group/role level.

Detail Description Paragraph - DETX (2465):
[2490] **Monitors** change to make sure that change is delivered on-time according to established plans, making adjustments to the plan when unforeseen issues or events arise (e.g., rollout management, change control, asset management etc.)

Detail Description Paragraph - DETX (2479):
[2504] Therefore, further paper forms are typically used for raising change requests but the change administrator then stores the most important information in a change request **database**. The decision will depend primarily on the size of the system.

Detail Description Paragraph - DETX (2494):
[2519] There are four options to consider, when designing the scope of the Asset Management function. Usage of the Asset inventory only as a production system **database** (core **database**), including hardware devices, software versions loaded in the production environment, their licenses and network configuration data. Thus the asset inventory system only stores the core systems components in the production environment.

Detail Description Paragraph - DETX (2495):
[2520] In addition to the production system data as describes above, it contains any existing release and release components such as software modules, documents and procedures. It also contains **service level agreements** and actual figures for user groups and devices, incidents, problems and change requests. It may also contain additional data such as performance data or log of all backups taken.

Detail Description Paragraph - DETX (2522):
[2547] Rollout Management is concerned with delivering new sites or services to existing sites on-time based on the rollout schedule. Rollout Management **monitors** the rollout progress of all functions against the rollout schedule to ensure that the schedule is maintained. Review of the rollout schedule takes place regularly to determine how well rollout is progressing and to make any adjustments to the rollout schedule based upon any problems or issues which arise.

Detail Description Paragraph - DETX (2532):
[2557] Rollout Planning handles the greatest period of change in distributed systems management--system rollout and installation. During rollout every site and every user may be impacted by the changes taking place. Since delivery of the system will affect how well it is received by the users and is oftentimes defined by an **SLA**(s), delivery of the system must take place smoothly with

minimal interruption to the users. This can be challenging when both old and new architecture domains must exist concurrently until the rollout has been completed.

Detail Description Paragraph - DETX (2535):
[2560] Release Control is concerned with delivering a release on-time based upon the release schedule. Release Control **monitors** the release progress of all activities against the schedule to ensure that the schedule is maintained. Review of the release schedule takes place regularly to determine how well the release is progressing and to make any adjustments to the release schedule based upon any issues or problems which arise.

Detail Description Paragraph - DETX (2550):
[2575] **Monitoring** and delivery of releases as well as review of release schedule versus planned schedule.

Detail Description Paragraph - DETX (2556):
[2581] Confirmation of release scheduling and determine if the release is on schedule and **report** on progress of release.

Detail Description Paragraph - DETX (2573):
[2598] There are four options to consider, when designing the scope of the Asset Management function. Usage of the Asset inventory only as a production system **database** (core **database**), including hardware devices, software versions loaded in the production environment, their licenses and network configuration data. Thus the asset inventory system only stores the core systems components in the production environment.

Detail Description Paragraph - DETX (2574):
[2599] In addition to the production system data as describes above, it contains any existing release and release components such as software modules, documents and procedures. It also contains **service level agreements** and actual figures for user groups and devices, incidents, problems and change requests. It may also contain additional data such as performance data or log of all backups taken.

Detail Description Paragraph - DETX (2600):
[2625] **Database** Management (1338)

Detail Description Paragraph - DETX (2601):
[2626] **Database** Management is the management and administration of **database** technologies, including **monitoring,** physical file placement, performance, and sizing.

Detail Description Paragraph - DETX (2602):

[2627] **Database** Recovery

Detail Description Paragraph - DETX (2603):
[2628] **Database** Recovery is the process of providing recovery of **database** entities following a logical or physical **database** failure. This includes **database** software failure and local disk failure.

Detail Description Paragraph - DETX (2604):
[2629] **Database** Disaster Recovery

Detail Description Paragraph - DETX (2605):
[2630] **Database** Disaster Recovery is the process of recovering the **database** entities following a catastrophic failure. This process should be fully integrated in the enterprise-wide disaster recovery plan.

Detail Description Paragraph - DETX (2606):
[2631] **Database** Backup/Restore Management

Detail Description Paragraph - DETX (2607):
[2632] **Database** Backup/Restore Management is the process of providing point-in-time backup and recovery for logical **database** restores. This includes application-driven data errors, dropped tables, and corrupt data.

Detail Description Paragraph - DETX (2609):
[2634] Capacity Modeling & Planning ensures that adequate **resources** will be in place to meet the **SLA** requirements, keeping in mind operational requirements which may require additional capacity. **Resources** can include such things as physical facilities, computers, memory/disk space, communications lines and personnel. Through this component, changes to the existing environment will be determined, modeled and planned according to the necessary requirements.

Detail Description Paragraph - DETX (2612):
[2637] Capacity Planning & Modeling must coordinate the requirements across the system (e.g., networks, servers, workstations, CPU, etc.) Capacity is driven by the need to meet **SLAs** with the user communities and as part of the planning and modeling process, future threats to capacity should be identified.

Detail Description Paragraph - DETX (2614):
[2639] **Monitoring** (1340)

Detail Description Paragraph - DETX (2617):
[2642] Performance Management ensures that the required **resources** are available at all times throughout the distributed system to meet the agreed upon **SLAs**. This includes **monitoring** and management of end-to-end performance

based on utilization, capacity, and overall performance statistics. If necessary, Performance Management can make adjustments to the production environment to either enhance performance or rectify degraded performance.

Detail Description Paragraph - DETX (2623):
[2648] Performance Management needs to consider performance from a business perspective, not merely a systems one. Most transactions in distributed systems utilize a wide variety of **resources,** and the measurement of end-to-end response time becomes the sum of the time expended by each one of the components sequentially involved in the transaction less the time while components were processing in parallel.

Detail Description Paragraph - DETX (2624):
[2649] What devices/users will be **monitored** and at which locations? Will this information change?

Detail Description Paragraph - DETX (2626):
[2651] Will performance be **measured** from end-to-end or merely for individual components?

Detail Description Paragraph - DETX (2628):
[2653] Will **monitoring** be continuous or by demand?

Detail Description Paragraph - DETX (2629):
[2654] Continuous **monitoring** can generate significant performance overhead, whereas targeted, periodic **monitoring** may only be necessary. This strategy will impact the design of the technical infrastructure as well as the tools chosen the manage the systems performance.

Detail Description Paragraph - DETX (2630):
[2655] Will only selected transactions be **measured,** and if so, should this selection be configurable?

Detail Description Paragraph - DETX (2631):
[2656] It may be necessary to **measure** business critical transactions only; specified within the **SLA**. If the facility to select specific transactions is required, significant customization of the system may be necessary.

Detail Description Paragraph - DETX (2633):
[2658] Once transaction have been selected for **monitoring,** the decision needs to be taken whether or not every transaction of that type should be **monitored,** or only a sample set of those transactions. Full **monitoring** may increase network and processing overheads.

Detail Description Paragraph - DETX (2635):
[2660] As **SLAs** will likely be tied in some way to performance, it is important to **monitor** and correct the systems performance as it degrades to ensure that operational levels are maintained and that the **SLA**(s) will not be violated.

Detail Description Paragraph - DETX (2638):
[2663] Collect, analyze and display in graphical format real-time performance characteristics from a wide range of **resources**. Analyze current workload and configuration data and forecast future requirements, as well as providing input into the Financial planning process.

Detail Description Paragraph - DETX (2640):
[2665] Provide real time **monitoring** and interactive tuning of the environment. Ability to input threshold alerting based on high/low watermarks and proactively act.

Detail Description Paragraph - DETX (2641):
[2666] **Monitoring** capabilities include the ability to **measure** CPU and disk utilization, memory occupancy, transaction response time, **reports** (storage & distribution), printers, network utilization and performance, circuit utilization, backup facilities, WAN/LAN utilization.

Detail Description Paragraph - DETX (2644):
[2669] May require use of some or all of the following **monitoring** tools: operating system **monitor,** on-line **monitor,** batch **monitor,** data base **monitor,** (host, server) and network **monitor** (WAN, LAN).

Detail Description Paragraph - DETX (2646):
[2671] Performance **measures** must be consistent with Service Level management techniques

Detail Description Paragraph - DETX (2648):
[2673] **Resource** utilization statistics may be used to generate costing, and potential billings for customers.

Detail Description Paragraph - DETX (2649):
[2674] Passes data to the **resource** consumption management facility to **report** on the recurring processing cost of each business application.

Detail Description Paragraph - DETX (2651):
[2676] Physical Site Management **monitors** the central and distributed sites environmental and regulatory levels. Physical Site Management ensures that adequate power, cooling facilities, fire suppression, etc. are provided and maintained to prevent system outages. When necessary, corrective actions are

issued and **monitored** according to pre-defined environmental control plans.

Detail Description Paragraph - DETX (2670):
[2695] To ensure that the Operability Principles have been satisfied, each release should, in principle, undergo a release test of a full business cycle (to show that Operations can run it) and full business volumes (to show that **SLA** targets can be achieved). These tests are, however, expensive in terms of dedicated hardware requirements, people, and elapsed time.

Detail Description Paragraph - DETX (2673):
[2698] Repositories contain all the management data generated or used during the management process. This includes historical data, capacity data, performance data, problem knowledge bases, asset **databases,** solution sets, and management information bases (MIBs). The repositories component interacts with the management applications, integration platform, supporting infrastructure, and presentation components. Again it is important to make sure that the other components of the operational architecture are compatible with the **database** tools.

Detail Description Paragraph - DETX (2690):
[2715] File Transfer and Control initiates and **monitors** files being transferred throughout the system as part of the business processing (e.g., nightly batch runs). File transfers may occur between any two or more devises within the system.

Detail Description Paragraph - DETX (2697):
[2722] Store and forward techniques can help reduce the contention for system **resources** during business hours. Store and forward can also reduce the amount of traffic in the system based upon the routing tables defined within the system. Instead of having one machine send the same file to multiple machines, for instance, a cascading forwarding mechanism can be used. This also improves the system performance as files are sent a minimal number of times to certain devices which then forward the files on to other devices.

Detail Description Paragraph - DETX (2701):
[2726] Managing hardware is all hardware directly used to manage the environment. This includes all staging components. These components are devoted to systems management functions. Examples of managing hardware include management servers, management controllers, management consoles, probes, and sniffers. One significant component in the hardware **monitoring** arena is Firewall access control policy management. Firewalls are regularly used for network based security management. It is typically a system or group of systems that enforce access control between two or more networks and/or perform network data packet filtering. Usually packet filtering router hardware and application gateways are used to block unauthorized IP packets and enforce proxy defined user commands.

Detail Description Paragraph - DETX (2705):
[2730] In the event of a significant system failure, Disaster Recovery processes will be invoked to re-route the system **resources** to a secondary, stable configuration until the primary **resources** can be restored. Within a distributed environment, disaster recovery must account for differing levels of disaster whether at a central or distributed site(s).

Detail Description Paragraph - DETX (2714):
[2739] Hardware Maintenance maintains all of the components within a distributed system to protect the investment of the organization. Generally agreed upon in the **SLAs,** maintenance contracts are carried out, **monitored** and recorded for each asset as appropriate.

Detail Description Paragraph - DETX (2720):
[2745] **Monitoring** (1358)

Detail Description Paragraph - DETX (2725):
[2750] What type of events will be **monitored**? More specifically, what services need to be **monitored** across which devices (e.g., servers, workstations, routers, hubs, bridges)?

Detail Description Paragraph - DETX (2726):
[2751] The scope of events to be **monitored** will have a major impact on the approach taken for Event management and the tools selected.

Detail Description Paragraph - DETX (2736):
[2761] Performance Management ensures that the required **resources** are available at all times throughout the distributed system to meet the agreed upon **SLAs**. This includes **monitoring** and management of end-to-end performance based on utilization, capacity, and overall performance statistics. If necessary, Performance Management can make adjustments to the production environment to either enhance performance or rectify degraded performance.

Detail Description Paragraph - DETX (2738):
[2763] Physical Site Management **monitors** the central and distributed sites environmental and regulatory levels. Physical Site Management ensures that adequate power, cooling facilities, fire suppression, etc. are provided and maintained to prevent system outages. When necessary, corrective actions are issued and **monitored** according to pre-defined environmental control plans.

Detail Description Paragraph - DETX (2764):
[2789] Will Equipment be **resourced** from multiple or single suppliers?

Detail Description Paragraph - DETX (2768):
[2793] **Monitoring** (1364)

Detail Description Paragraph - DETX (2771):

[2796] Physical Site Management **monitors** the central and distributed sites environmental and regulatory levels. Physical Site Management ensures that adequate power, cooling facilities, fire suppression, etc. are provided and maintained to prevent system outages. When necessary, corrective actions are issued and **monitored** according to pre-defined environmental control plans.

Claims Text - CLTX (2):

1. A method for designing, implementing, and maintaining a development architecture framework comprising the steps of: (a) managing information that supports a project being carried out by a development architecture framework; (b) handling security of the development architecture framework by defining security requirements and auditing the development architecture framework to ensure that the security requirements are met; (c) ensuring quality of the project being carried out by the development architecture framework by obtaining measurements relating to predetermined criterion of the project, statistically analyzing the measurements, and training personnel based on the statistical analysis in order to improve the quality of the project; (d) managing the project being carried out by the development architecture framework by generating a plan to carry out the project, scheduling a timeline for executing the plan, tracking the. execution of the plan, and **reporting** information uncovered during tracking; (e) governing an environment in which the project is carried out by the development architecture framework; (f) coordinating the delivery of components of the project in a selected order; (g) rectifying problems that occur during the delivery of the components of the project; and (h) maintaining updated support information during the delivery of the components of the project.

Claims Text - CLTX (5):

4. The method as set forth in claim 3, wherein the collaboration tools are selected from the group of collaboration tools including audio/video conferencing, electronic mail, group scheduling, shared workspaces, and shared **databases**.

Claims Text - CLTX (9):

8. A computer program embodied on a computer readable medium for designing, implementing, and maintaining a development architecture framework comprising: (a) a code segment that manages information that supports a project being carried out by a development architecture framework; (b) a code segment that handles security of the development architecture framework by defining security requirements and auditing the development architecture framework to ensure that the security requirements are met; (c) a code segment that ensures quality of the project being carried out by the development architecture framework by obtaining measurements relating to predetermined criterion of the project, statistically analyzing the measurements, and training personnel based on the statistical analysis in order to improve the quality of the project; (d) a code segment that manages the project being carried out by the development architecture framework by generating a plan to carry out the project,

scheduling a timeline for executing the plan, tracking the execution of the plan, and **reporting** information uncovered during tracking; (e) a code segment that governs an environment in which the project is carried out by the development architecture framework; (f) a code segment that coordinates the delivery of components of the project in a selected order; (g) a code segment that rectifies problems that occur during the delivery of the components of the project; and (h) a code segment that maintains updated support information during the delivery of the components of the project.

Claims Text - CLTX (12):

11. The computer program as set forth in claim 10, wherein the collaboration tools are selected from the group of collaboration tools including audio/video conferencing, electronic mail, group scheduling, shared workspaces, and shared **databases**.

Claims Text - CLTX (16):

15. A system for designing, implementing, and maintaining a development architecture framework comprising: (a) logic for managing information that supports a project being carried out by a development architecture framework; (b) logic for handling security of the development architecture framework by defining security requirements and auditing the development architecture framework to ensure that the security requirements are met; (c) logic for ensuring quality of the project being carried out by the development architecture framework by obtaining measurements relating to predetermined criterion of the project, statistically analyzing the measurements, and training personnel based on the statistical analysis in order to improve the quality of the project; (d) logic for managing the project being carried out by the development architecture framework by generating a plan to carry out the project, scheduling a timeline for executing the plan, tracking the execution of the plan, and **reporting** information uncovered during tracking; (e) logic for governing an environment in which the project is carried out by the development architecture framework; (f) logic for coordinating the delivery of components of the project in a selected order; (g) logic for rectifying problems that occur during the delivery of the components of the project; and (h) logic for maintaining updated support information during the delivery of the components of the project.